Fall 2017: CSci-4131 Internet Programming
Assignment 6

Issued: Sunday Nov 12, 2017

Due Date and Time: Sunday November 26 at 11:55pm

Task:
In this assignment you will create a login capability for your Calendar Website and learn about how to restrict the access to pages of your Calendar website by using a relational database (MySQL).

This is a very practical php application that is widely used in web portals and content management systems (CMS) like WordPress, and PHP handles it very well.

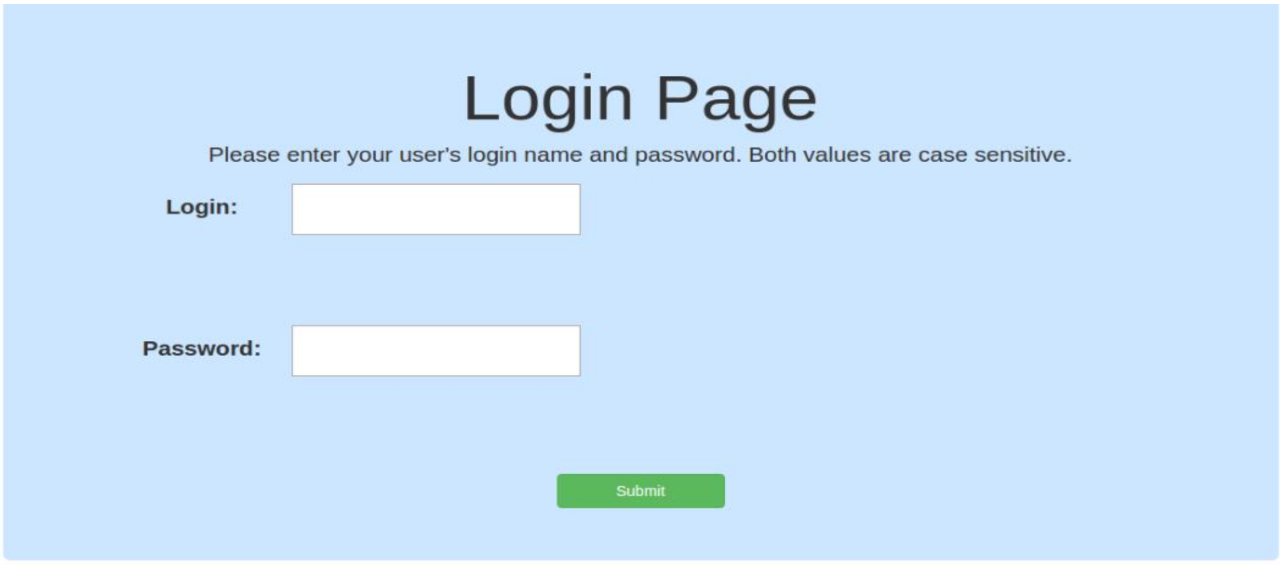
You must create two pages that are critical for your task: ***login.php***, and ***logout.php***.
If a user attempts to use your website, but is not logged in, he/she should be redirected to the login page. The login page gets and validates the user login ID and password, checks the user login ID against the database, checks the password against the database, and if everything matches, redirects the user to the first page of your website - the calendar page.

From there, the user may navigate to the second page of your website, where the *Input form* is present. The pages of your website need to be protected from anyone who does not have a login and password in your system. You can use **$_SESSION** variable in PHP to store the current user's information, and if it is not set, redirect the person attempting to use the Calendar or Input page to the login page.

Functionality:

- If user is not authenticated, opening calendar page or Form input page should redirect user to ***login.php***
- ***login.php*** should display a login form with at least the following:
  - ➢ *Login ID Field*
  - ➢ *Password Field*
  - ➢ *Submit Button*
  - ➢ *Div for Errors*

The following pages illustrate and describe the behavior you will implement when someone attempts to access your Calendar or Form php scripts. First, if a person attempts to run your Calendar.php or Form.php scripts and they are not “logged in”, the login page will be displayed as follows.

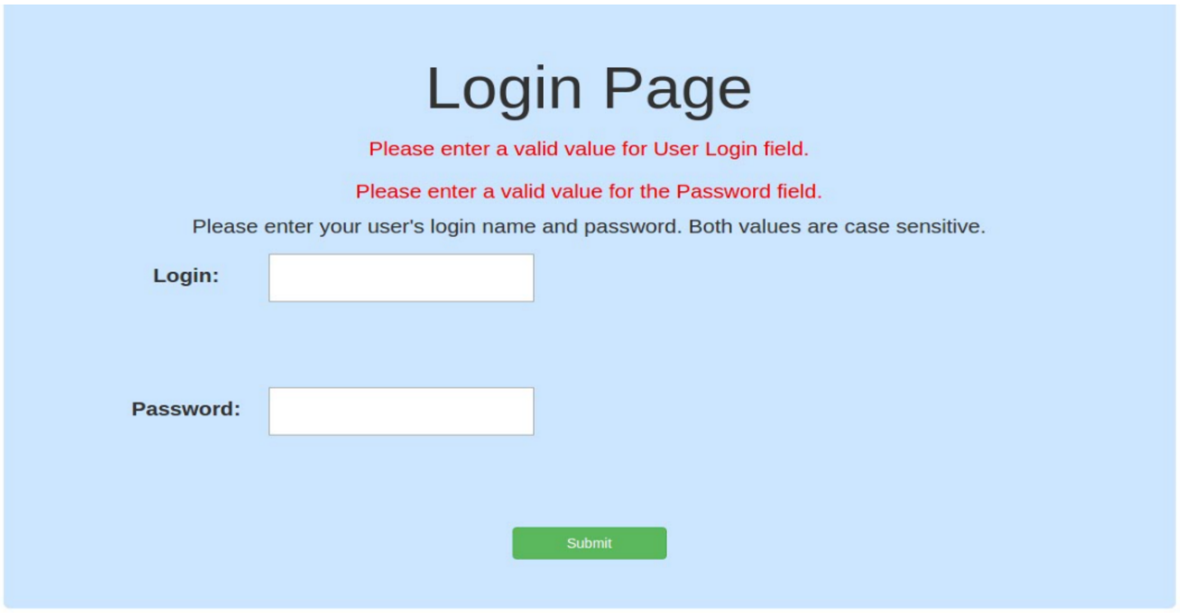

Login Page

Please enter your user's login name and password. Both values are case sensitive.

Login:

Password:

- When the submit button is clicked, your login page should validate the input and, if necessary, display the following errors:
  - ➢ *Please enter a valid value for Login Name Field.*
  - ➢ *Please enter a valid value for Password Field.*
  - ➢ *or both, if both Fields are empty*


Login Page

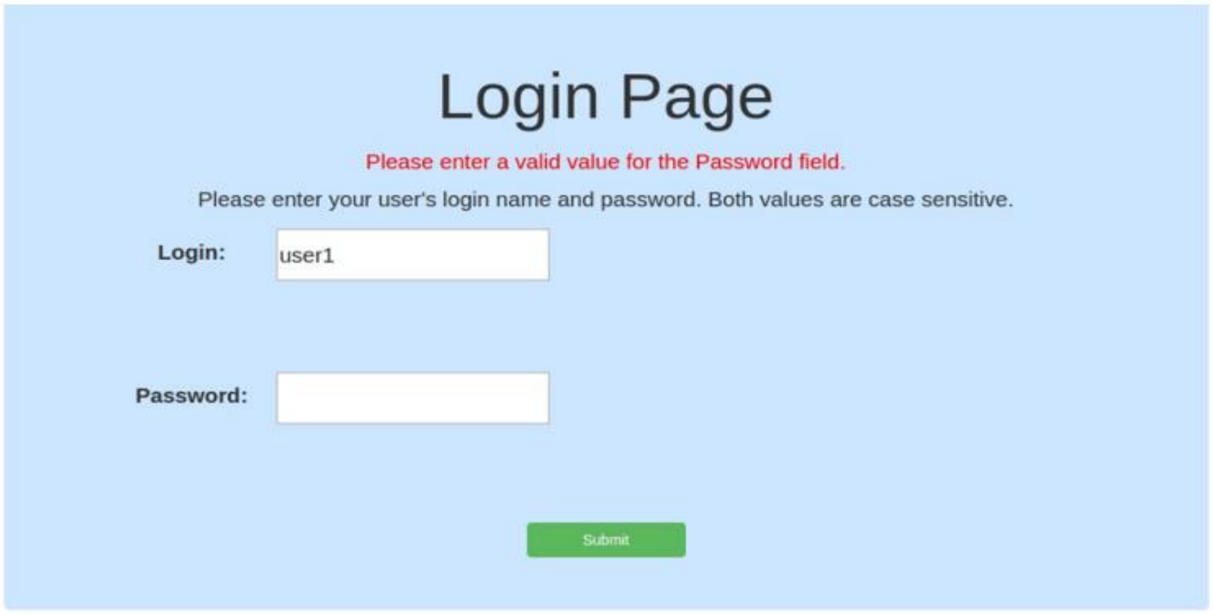
Please enter a valid value for User Login field.

Please enter a valid value for the Password field.

Please enter your user's login name and password. Both values are case sensitive.

Login:

Password:


Login Page

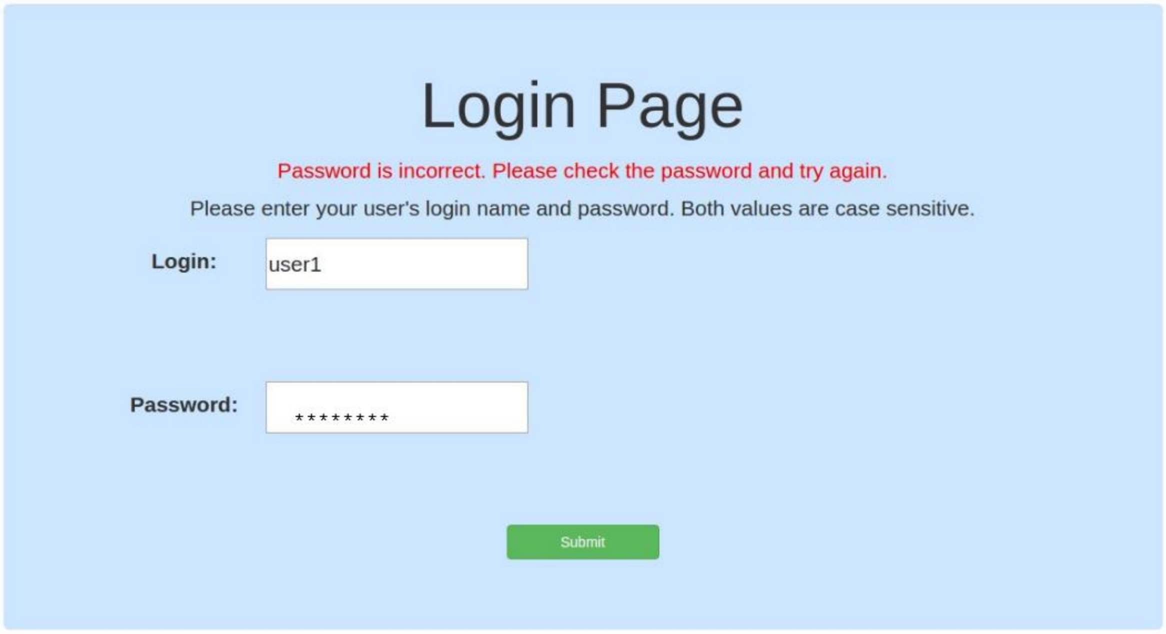
Please enter a valid value for the Password field.

Please enter your user's login name and password. Both values are case sensitive.

Login: user1

Password:

- After your login script successfully obtains values for the login and password, your login script should query the database to check the values entered against the values stored in the database. The following errors need to be reported should they occur:
  - ***Login is incorrect: User does not exist. Please check the login details and try again.*** (This will be displayed if the Login ID entered is not present in the database users list)
  - ***Password is incorrect: Please check the password and try again.*** (This will be displayed if the correct login is entered, but the encrypted password associated with the Login ID in the database is not the same as the encrypted password entered by the user.)


Login Page

Password is incorrect. Please check the password and try again.

Please enter your user's login name and password. Both values are case sensitive.

Login: user1

Password: ********

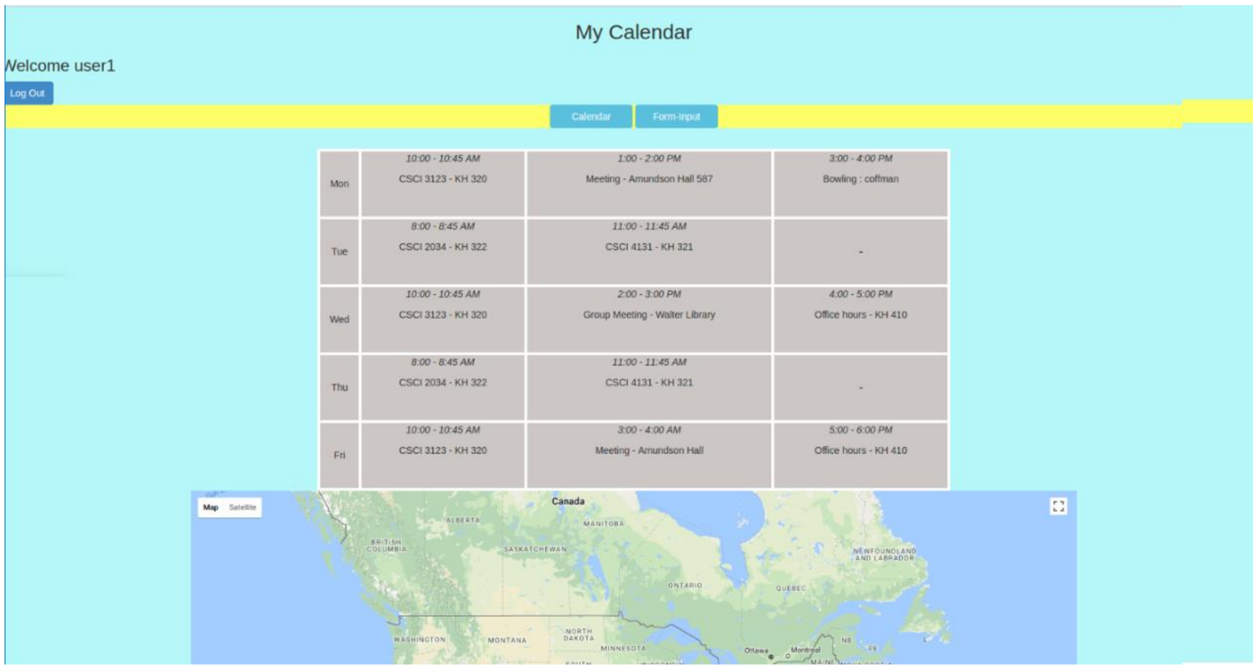
- Passwords should be stored in the database in a SHA1 hashed format. You need to convert the password string entered by the user into a SHA1 hash format using sha1() function of PHP, and compare it to the SHA1 hashed password stored in the database. You should edit and run the **insertValues** PhP script provided to insert a user or users, login names and passwords into your database before running your login script.

- When a user has successfully logged in, the *Calendar page* and *Form Ipage* should display a welcome message with the current user name, and display a logout button or link.

- When a user selects the *LogOut* button on the Calendar or Form Pages, your script should should clear the session variable, destroy the session, and redirect the user to *login.php* page.

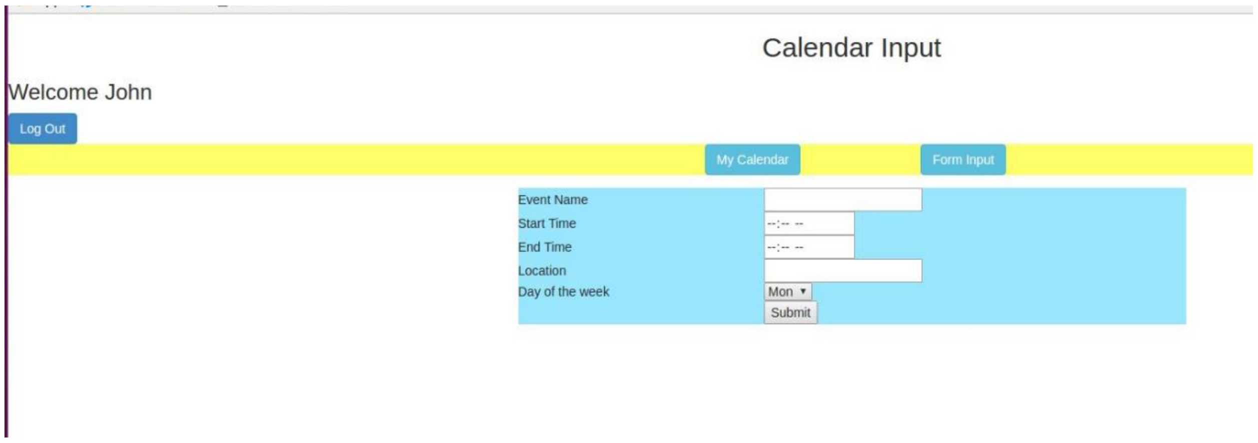
- Once logged out, a user can no longer open the *Calendar* or *Form* pages without successfully re-entering her or his credentials via the *login.php* page.

MyCalendar.php



Mon	*10:00 - 10:45 AM* CSCI 3123 - KH 320	*1:00 - 2:00 PM* Meeting - Amundson Hall 587	*3:00 - 4:00 PM* Bowling : coffman
Tue	*8:00 - 8:45 AM* CSCI 2034 - KH 322	*11:00 - 11:45 AM* CSCI 4131 - KH 321	-
Wed	*10:00 - 10:45 AM* CSCI 3123 - KH 320	*2:00 - 3:00 PM* Group Meeting - Walter Library	*4:00 - 5:00 PM* Office hours - KH 410
Thu	*8:00 - 8:45 AM* CSCI 2034 - KH 322	*11:00 - 11:45 AM* CSCI 4131 - KH 321	-
Fri	*10:00 - 10:45 AM* CSCI 3123 - KH 320	*3:00 - 4:00 AM* Meeting - Amundson Hall	*5:00 - 6:00 PM* Office hours - KH 410

MyForm.php



- In the Assignment files for this assignment, you will find 3 skeleton files:
    - **createTables_HW6F17.php**;
    - **insertValues_HW6F17.php**, and
    - **database_HW6F17.php**.

- The **database_HW6F17.php** file contains code that sets php variables which contain values you can use to create a connection to your MYSQL database.

- Edit the file and add your database name, database username, and password credentials as provided to you in the column named **Database Information** in your grades on Moodle. Your Database name and password are provided (password is in the feedback column). Note, your database name and database user name are the same (F17CS4131Uxxx, where xxx is a number between 1 and 130)

- Edit the other two files and add your MySQL account credentials as you did with the **database_HW6F17.php** file, and then execute the files to create the tables and add users with their sha-1 hashed passwords. You should use the php **include** or **include_once** statements in your login.php code to get the database server credentials from **database_HW6F17.php,** and to use those credentials (values stored in PHP variables) to execute PhP **mysqli** functions and methods to create your connection to the MYSQL database in your PhP code.

Database:

- The Database in this assignment contains one table*: tbl_accounts*
- The table (*tbl_accounts*) has following fields: *acc_id*, *acc_name*, *acc_login*, and *acc_password*.
- You can use following PHP code to initiate a database connection:

```
<?php
//…

include once 'database_HW6F17.php';

// Create connection
$conn=new mysqli($db_servername,$db_username,$db_password,$db_name,$db_port);
if ( $conn->connect_error ) {

      // report error

} else {

      // setup your query
}
//…
?>
```

To connect to the database directly from cselabs unix machines (you can issue SQL commands directly to the database via the command line) you can do the following:

```
% module load soft/mysql
% mysql – u <yourDBusername> -hcse-curly.cse.umn.edu -P3306 -p <yourDBname>
```

```
And then enter your password when prompted.
```

Once you have started MySQL, you can, and should, change your database password with the following command

```
set password=password('whateveryouwant');
```

Submission Instructions

Include the following files in one zipped file for your submission:

- login.php : Main Login page
- logout.php : Logout page
- The ***Calendar*** and ***Form input*** pages in your website with logout links, and modified to prevent access if a user has not successfully logged in (i.e., established a session).
- The modified **insertValues_HW6F17 php** file (which contains your passwords)
- style.css: (optional)
- A readme file with any additional details
- Any other files that you think we will need

Grading Criteria (100 points):

1. Submission instructions are met - **10 points**
2. Both the pages of your website redirect the user to login.php page automatically before authentication - **10 Points**
3. The Login page shows the form elements and submit button - **10 points**
4. The Login page handles input validation, checks and retrieves the user from the database, and displays the appropriate errors - **15 points**
5. The Login page logs the user in and redirects to calendar page - **15 points**
6. Both pages of your website display the appropriate user name at the top - **15 points**
7. It is possible to Navigate between pages and the pages are functional – **10 points**
8. The Logout functionality works correctly - **15 points**